Problem 8: Foveated Rendering

Points: 15

Author: Gary Hoffmann, Denver, Colorado, United States

Problem Background

Virtual Reality has exploded into the market in the last five years, being used for everything from games and entertainment to product design and engineering. One of the more recent advances in VR headset design is the addition of eye tracking to increase performance.

The human eye has an extremely narrow field of view in which perfect 20/20 vision is attainable and fine detail can be distinguished. This clarity of vision is due to the fovea, a small depression in the inner retina specialized for this purpose. However, due to the size of the fovea, the human eye can only see clearly within a field of view of less than 10°. The rest of our vision comes from the brain piecing together imagery as we look around.

Due to this fact, a VR headset only needs to render the highest resolution imagery directly where the user is looking. Images outside of that field of view can be rendered at a lower quality, increasing the performance of the system.

Problem Description

You have been tasked with writing a module for a virtual reality application that determines the rendering quality for each section of the headset's screen. For simplicity, your module will only deal with a single eye on a single screen. The screen will be

divided into a 20-by-20 grid of blocks.

Your program will be given the coordinates within the grid at which the user is currently focusing their sight, and will need to output the rendering level of each cell in the grid row by row.

The cell the user is looking directly at should be rendered at full quality - 100%. All cells around that cell should be rendered at half quality (50%), and all cells around those should be rendered at one-quarter quality (25%). All other cells should be rendered at the minimum level of 10%.

Page 2

For example, if the user is looking at the block in row 7, column 10, the rendering quality for each block in the grid would be:

	0	⋯	7	8	9	10	11	12	13	⋯	19
0	10%	⋯	10%	10%	10%	10%	10%	10%	10%	⋯	10%
⋮	⋮	⋱	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋰	⋮
4	10%	⋯	10%	10%	10%	10%	10%	10%	10%	⋯	10%
5	10%	⋯	10%	25%	25%	25%	25%	25%	10%	⋯	10%
6	10%	⋯	10%	25%	50%	50%	50%	25%	10%	⋯	10%
7	10%	⋯	10%	25%	50%	100%	50%	25%	10%	⋯	10%
8	10%	⋯	10%	25%	50%	50%	50%	25%	10%	⋯	10%
9	10%	⋯	10%	25%	25%	25%	25%	25%	10%	⋯	10%
10	10%	⋯	10%	10%	10%	10%	10%	10%	10%	⋯	10%
⋮	⋮	⋰	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋱	⋮
19	10%	⋯	10%	10%	10%	10%	10%	10%	10%	⋯	10%

Sample Input

The first line of your program's input, **received from the standard input channel**, will contain a positive integer representing the number of test cases. Each test case will include a single line of input containing two integers, separated by spaces, representing

the row and column number of the eye position within the screen's grid, respectively. Row and column numbers will be between 0 and 19 inclusive.

```
2
7 10
0 0
```

Sample Output

For each test case, your program must output the rendering quality percentage for each block in the grid. Each row should be printed as a separate line, and columns should be separated by spaces.

```
10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10
10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10
10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10
10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10
10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10
10 10 10 10 10 10 10 10 25 25 25 25 25 10 10 10 10 10 10 10
10 10 10 10 10 10 10 10 25 50 50 50 25 10 10 10 10 10 10 10
10 10 10 10 10 10 10 10 25 50 100 50 25 10 10 10 10 10 10 10
10 10 10 10 10 10 10 10 25 50 50 50 25 10 10 10 10 10 10 10
10 10 10 10 10 10 10 10 25 25 25 25 25 10 10 10 10 10 10 10
```



```
10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10
10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10
10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10
10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10
10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10
10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10
10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10
10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10
10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10
10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10
100 50 25 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10
50 50 25 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10
```

```
25 25 25 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10
10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10
10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10
10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10
10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10
10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10
10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10
10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10
10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10
10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10
10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10
10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10
10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10
10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10
10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10
10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10
10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10
10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10
```